Fatima Jinnah Women University

Opening Portals of Excellence Through Higher Education

“LAB 11”

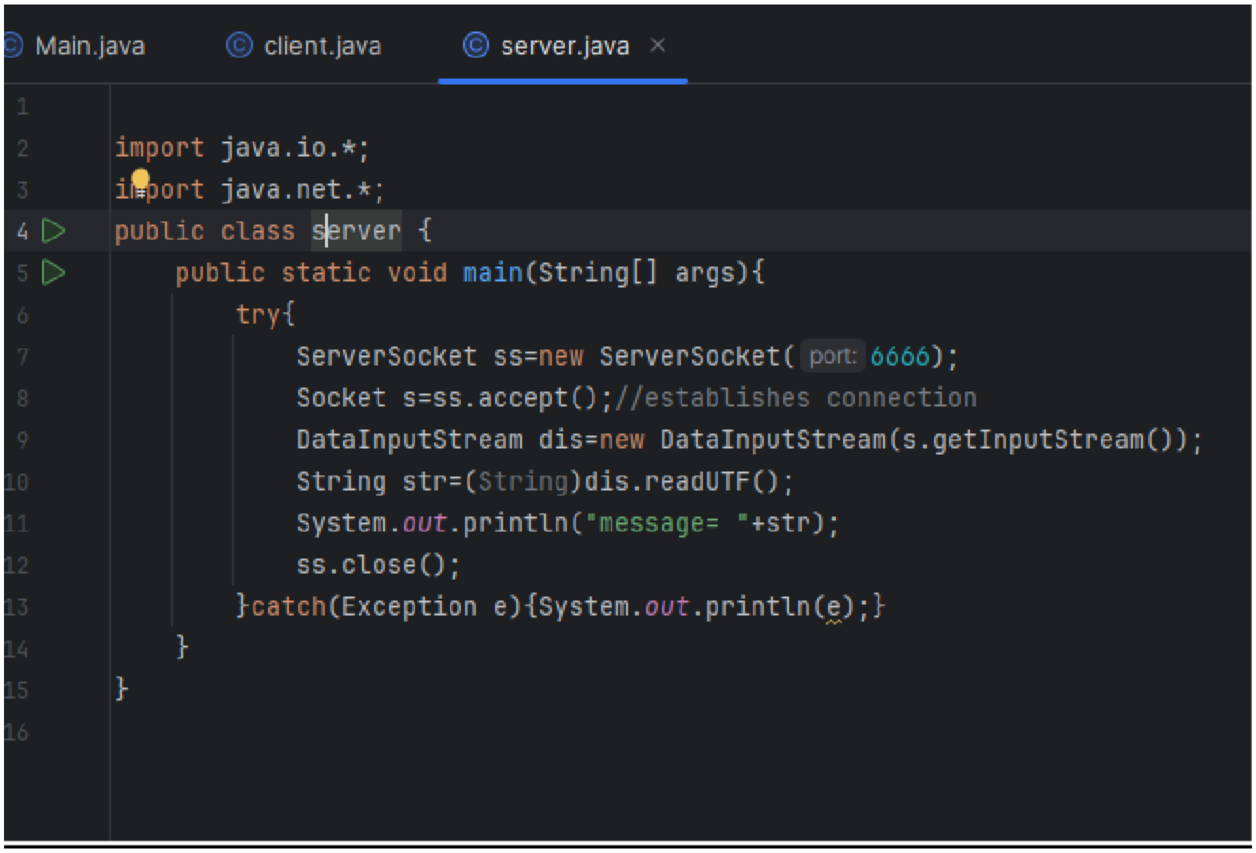
TANZEELA ASGHAR

2021-BSE-032 “A”

SUBMITTED TO: SIR SHAHZAD

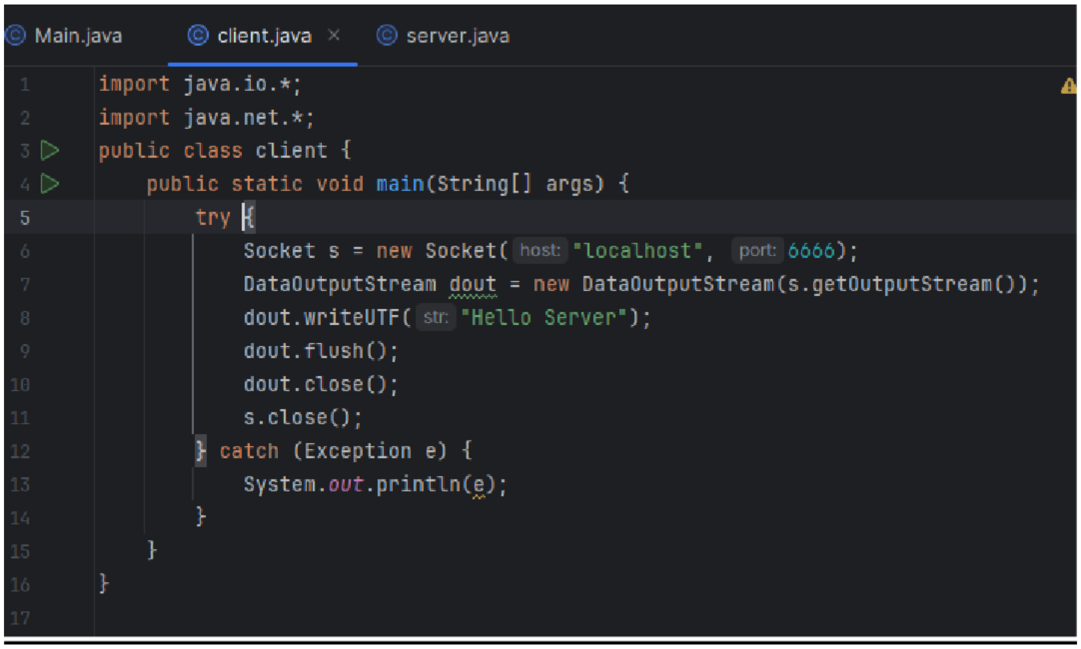
EXAMPLE 1

SERVER



```java

import java.io.*;
import java.net.*;
public class server {
    public static void main(String[] args){
        try{
            ServerSocket ss=new ServerSocket( port: 6666);
            Socket s=ss.accept();//establishes connection
            DataInputStream dis=new DataInputStream(s.getInputStream());
            String str=(String)dis.readUTF();
            System.out.println("message= "+str);
            ss.close();
        }catch(Exception e){System.out.println(e);}
    }
}

```

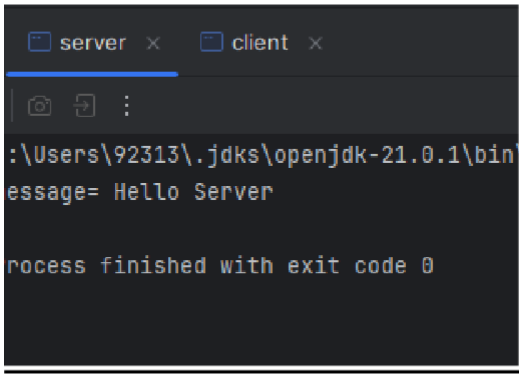
CLIENT



Main.java client.java server.java

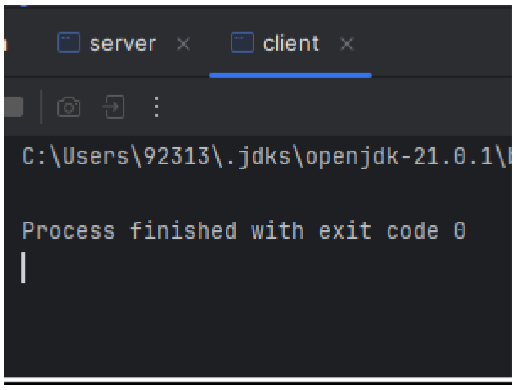
```java
import java.io.*;
import java.net.*;
public class client {
    public static void main(String[] args) {
        try {
            Socket s = new Socket( host: "localhost",  port: 6666);
            DataOutputStream dout = new DataOutputStream(s.getOutputStream());
            dout.writeUTF( str: "Hello Server");
            dout.flush();
            dout.close();
            s.close();
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

OUTPUT



server client

```
:\Users\92313\.jdks\openjdk-21.0.1\bin
essage= Hello Server

rocess finished with exit code 0
```

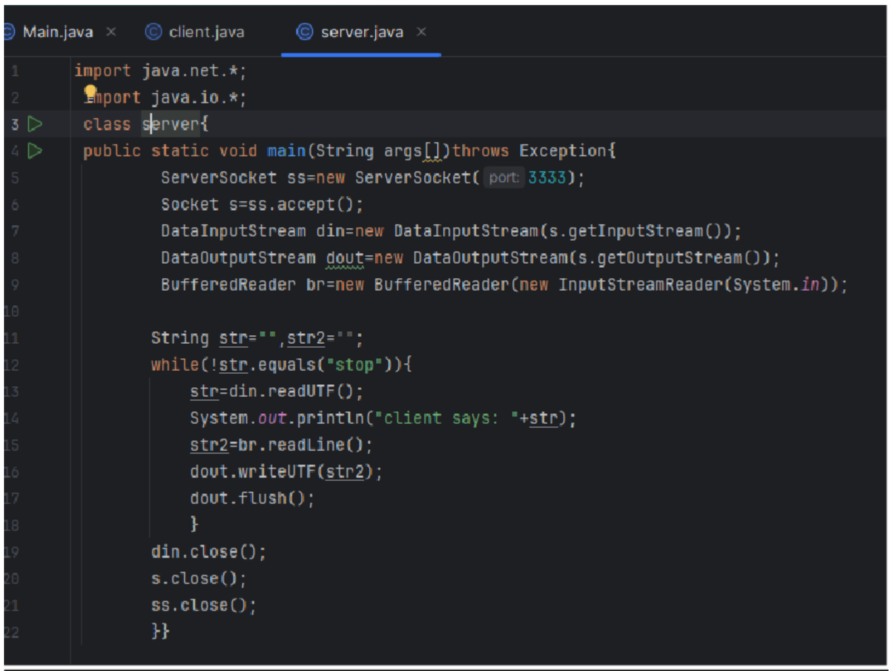


server client

```
C:\Users\92313\.jdks\openjdk-21.0.1\

Process finished with exit code 0
```

EXAMPLE 2

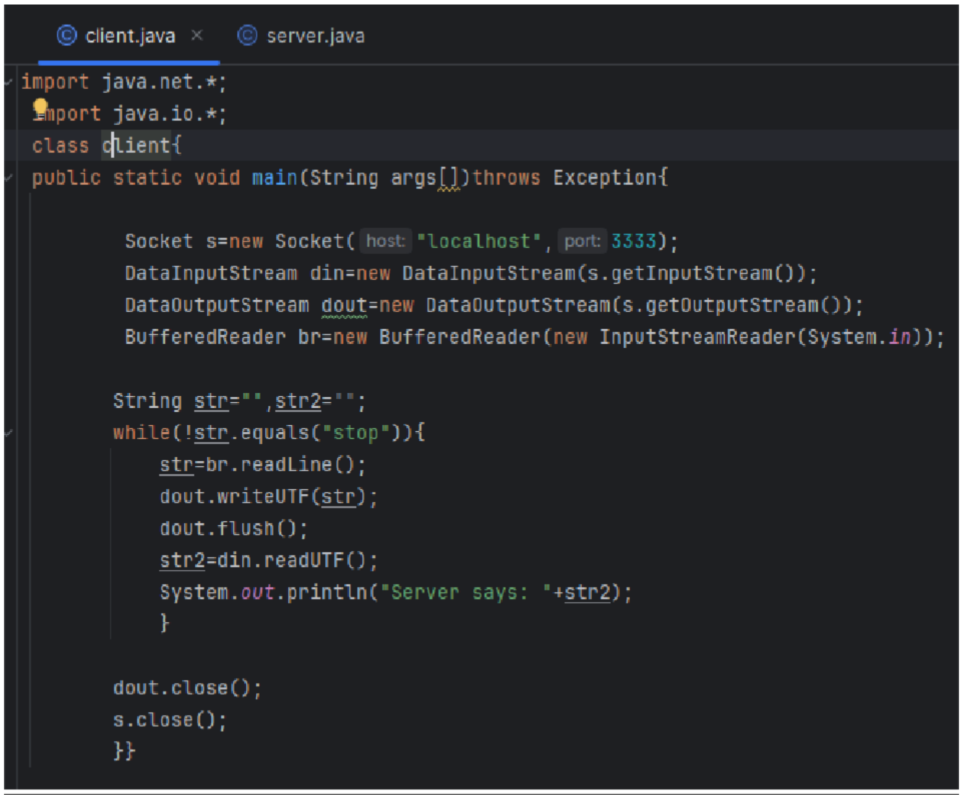
SERVER



Main.java × client.java server.java ×

```
import java.net.*;
 import java.io.*;
 class server{
 public static void main(String args[])throws Exception{
         ServerSocket ss=new ServerSocket( port: 3333);
         Socket s=ss.accept();
         DataInputStream din=new DataInputStream(s.getInputStream());
         DataOutputStream dout=new DataOutputStream(s.getOutputStream());
         BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

        String str="",str2="";
        while(!str.equals("stop")){
            str=din.readUTF();
            System.out.println("client says: "+str);
            str2=br.readLine();
            dout.writeUTF(str2);
            dout.flush();
            }
        din.close();
        s.close();
        ss.close();
        }}
```

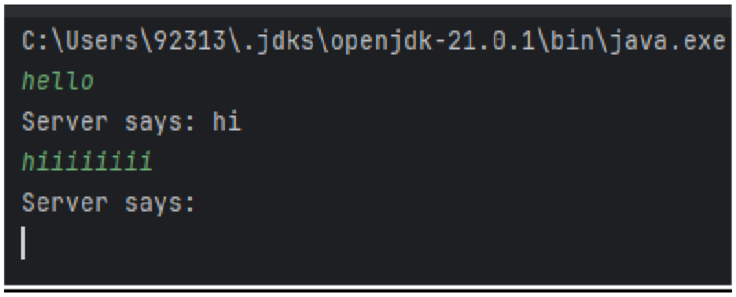
<u>CLIENT</u>



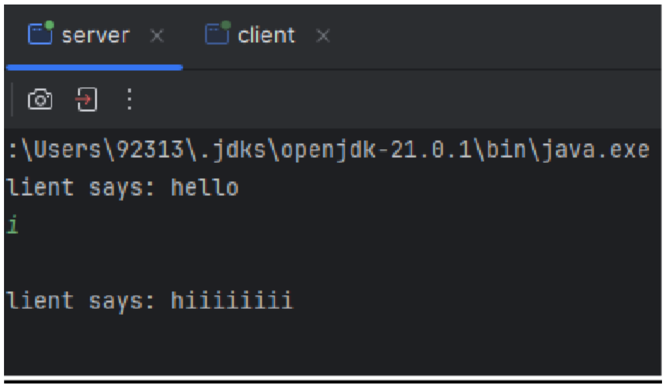
client.java × server.java

```
import java.net.*;
 import java.io.*;
 class client{
 public static void main(String args[])throws Exception{

         Socket s=new Socket( host: "localhost", port: 3333);
         DataInputStream din=new DataInputStream(s.getInputStream());
         DataOutputStream dout=new DataOutputStream(s.getOutputStream());
         BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

        String str="",str2="";
        while(!str.equals("stop")){
            str=br.readLine();
            dout.writeUTF(str);
            dout.flush();
            str2=din.readUTF();
            System.out.println("Server says: "+str2);
            }

        dout.close();
        s.close();
        }}
```

<u>OUTPUT</u>



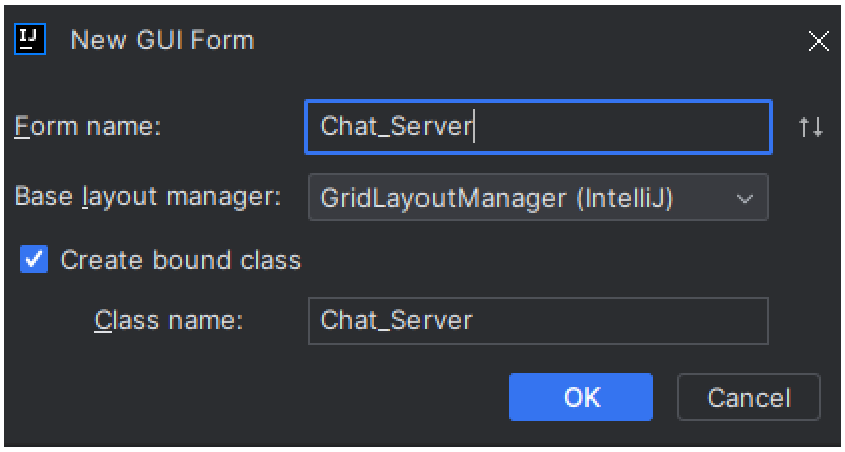
```
C:\Users\92313\.jdks\openjdk-21.0.1\bin\java.exe
hello
Server says: hi
hiiiiiiii
Server says:
```



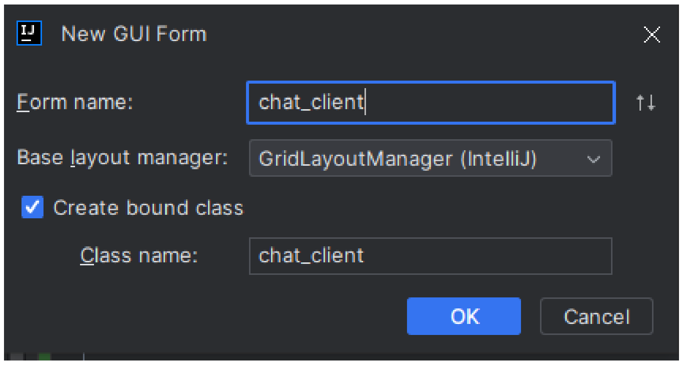
```
:\Users\92313\.jdks\openjdk-21.0.1\bin\java.exe
lient says: hello
i

lient says: hiiiiiiii
```

TASK:

BUILD CHAT APPLICATION USING SOCKET,SWING



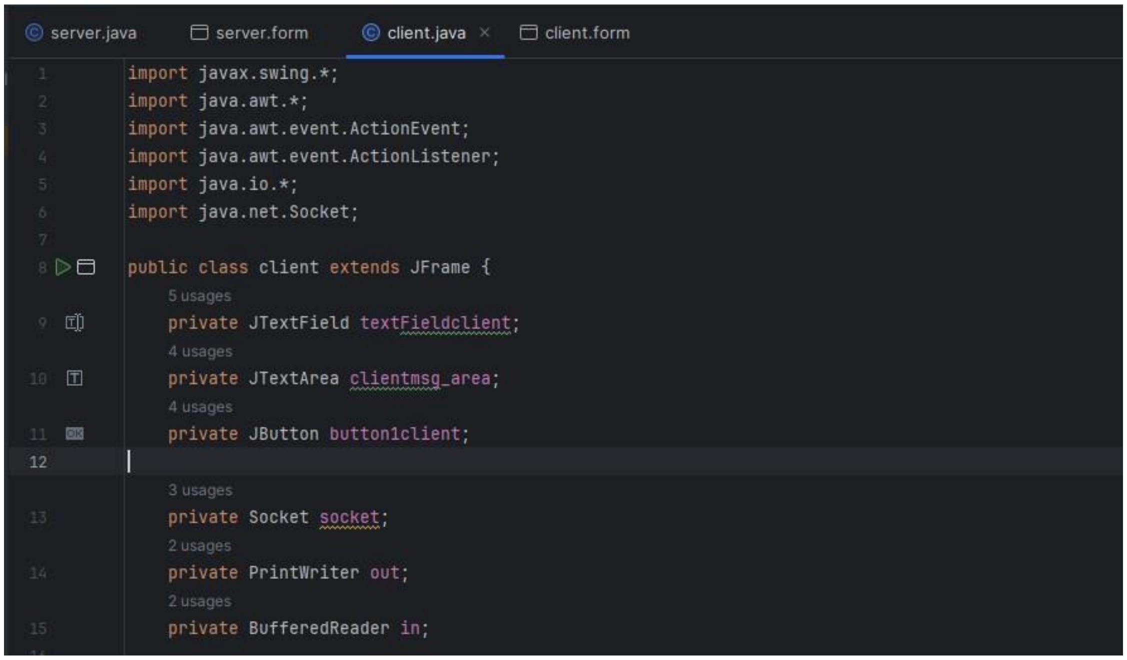
Form name: Chat_Server
Base layout manager: GridLayoutManager (IntelliJ)
Create bound class ✓
Class name: Chat_Server



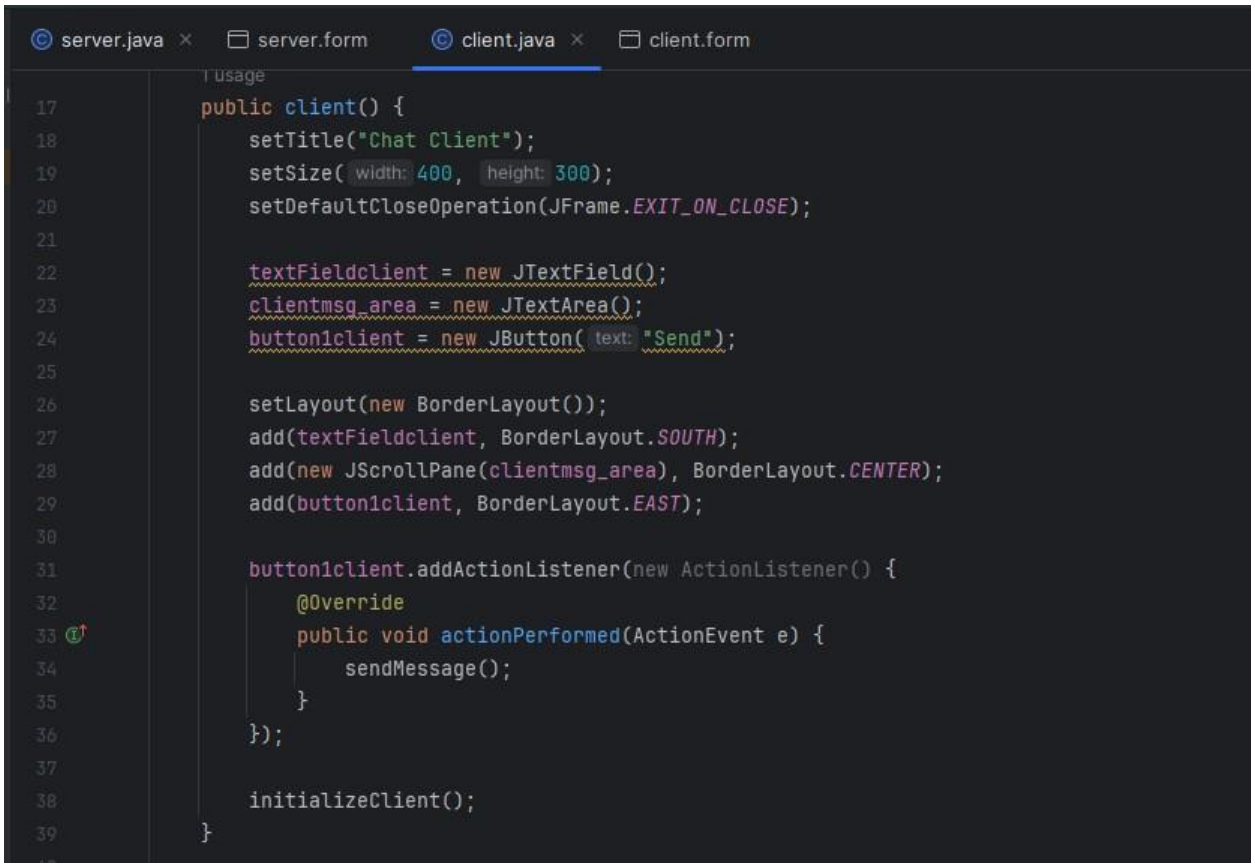
Form name: chat_client
Base layout manager: GridLayoutManager (IntelliJ)
Create bound class ✓
Class name: chat_client

CODE:

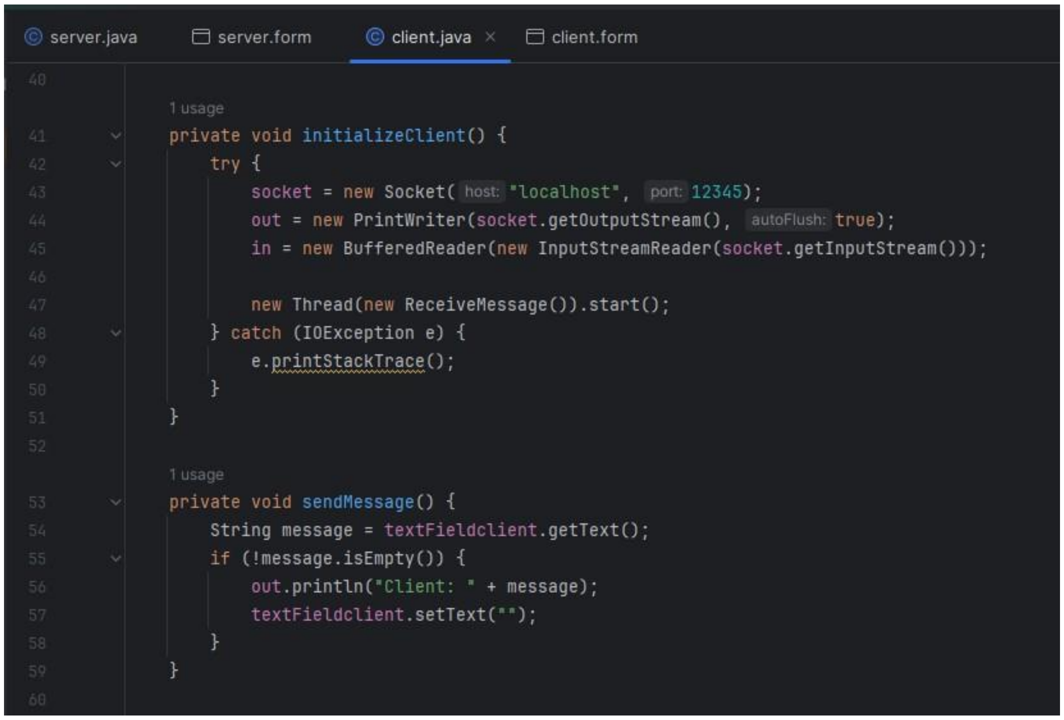
CLASS: CLIENT.JAVA



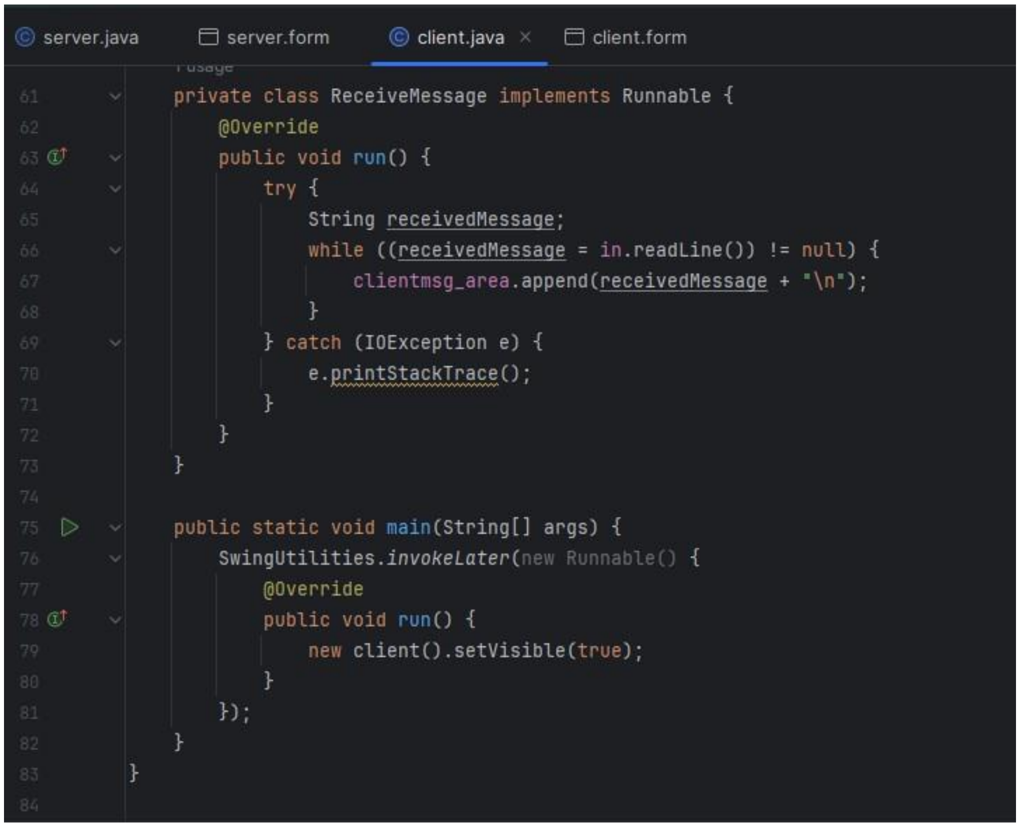
```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.net.Socket;

public class client extends JFrame {
    5 usages
    private JTextField textFieldclient;
    4 usages
    private JTextArea clientmsg_area;
    4 usages
    private JButton button1client;

    3 usages
    private Socket socket;
    2 usages
    private PrintWriter out;
    2 usages
    private BufferedReader in;
```



```java
    1 usage
    public client() {
        setTitle("Chat Client");
        setSize( width: 400,  height: 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        textFieldclient = new JTextField();
        clientmsg_area = new JTextArea();
        button1client = new JButton( text: "Send");

        setLayout(new BorderLayout());
        add(textFieldclient, BorderLayout.SOUTH);
        add(new JScrollPane(clientmsg_area), BorderLayout.CENTER);
        add(button1client, BorderLayout.EAST);

        button1client.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                sendMessage();
            }
        });

        initializeClient();
    }
```

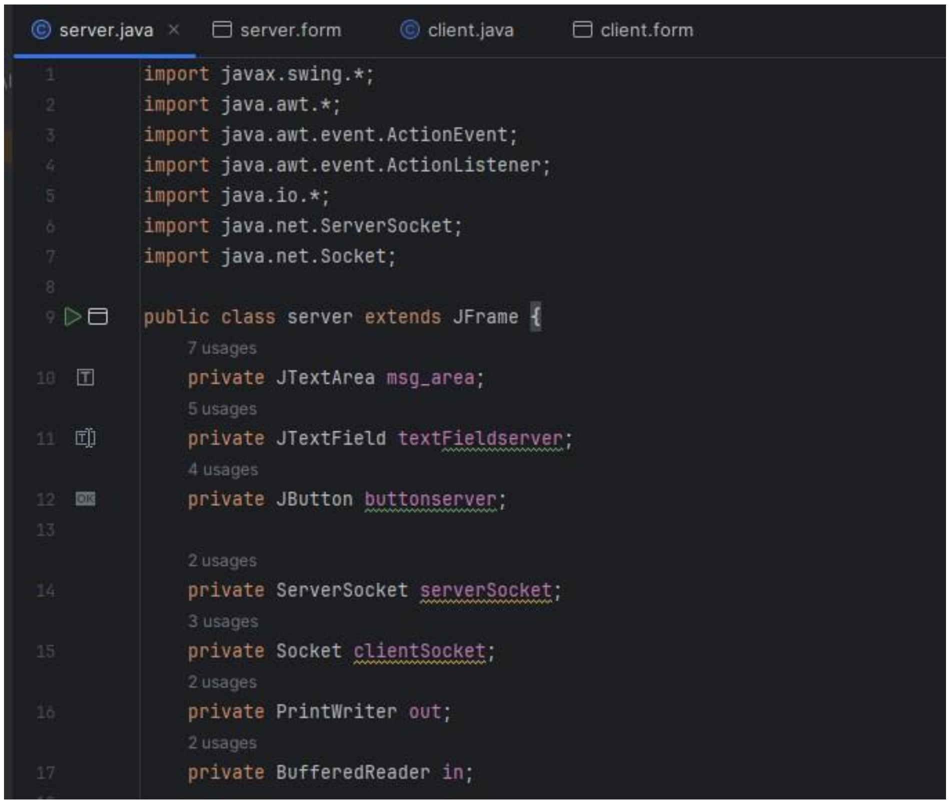


```java
    1 usage
    private void initializeClient() {
        try {
            socket = new Socket( host: "localhost",  port: 12345);
            out = new PrintWriter(socket.getOutputStream(),  autoFlush: true);
            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

            new Thread(new ReceiveMessage()).start();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    1 usage
    private void sendMessage() {
        String message = textFieldclient.getText();
        if (!message.isEmpty()) {
            out.println("Client: " + message);
            textFieldclient.setText("");
        }
    }

```

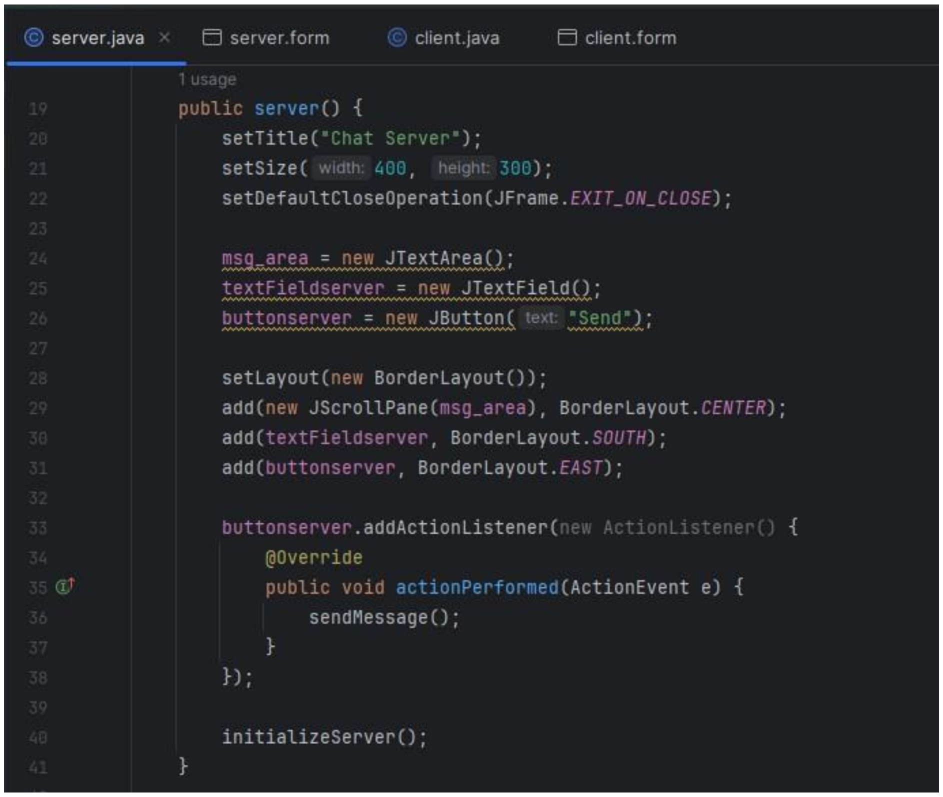


```java
    private class ReceiveMessage implements Runnable {
        @Override
        public void run() {
            try {
                String receivedMessage;
                while ((receivedMessage = in.readLine()) != null) {
                    clientmsg_area.append(receivedMessage + "\n");
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                new client().setVisible(true);
            }
        });
    }
}

```

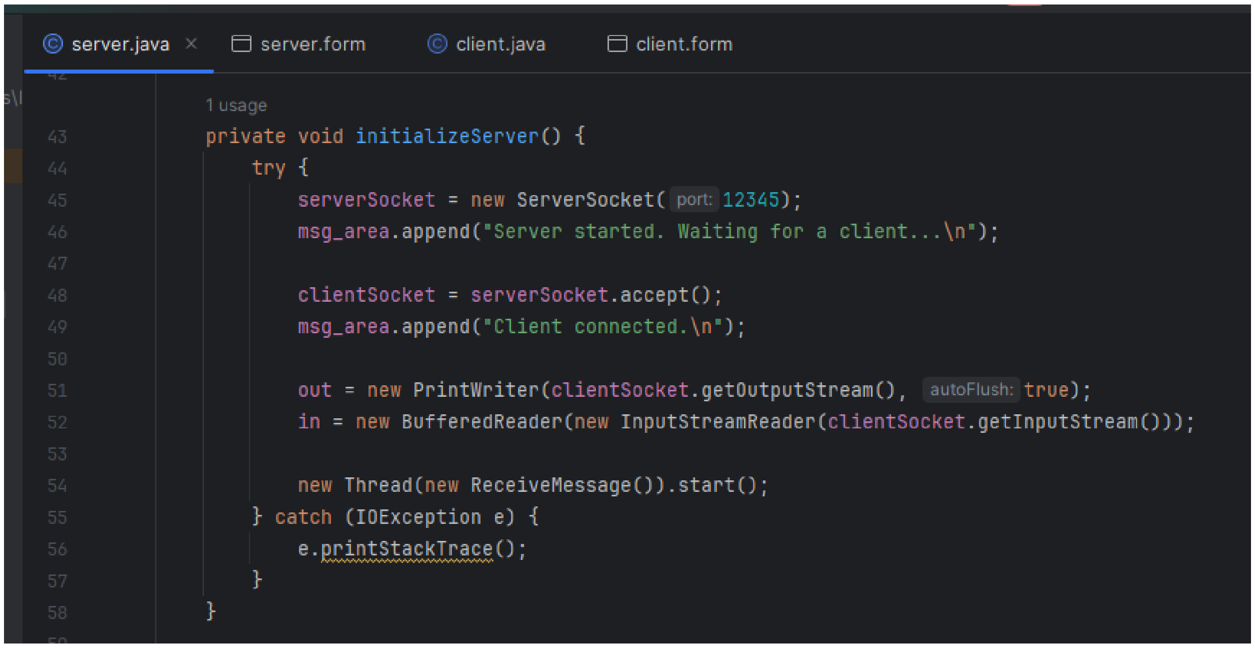
CLASS: SERVER.JAVA



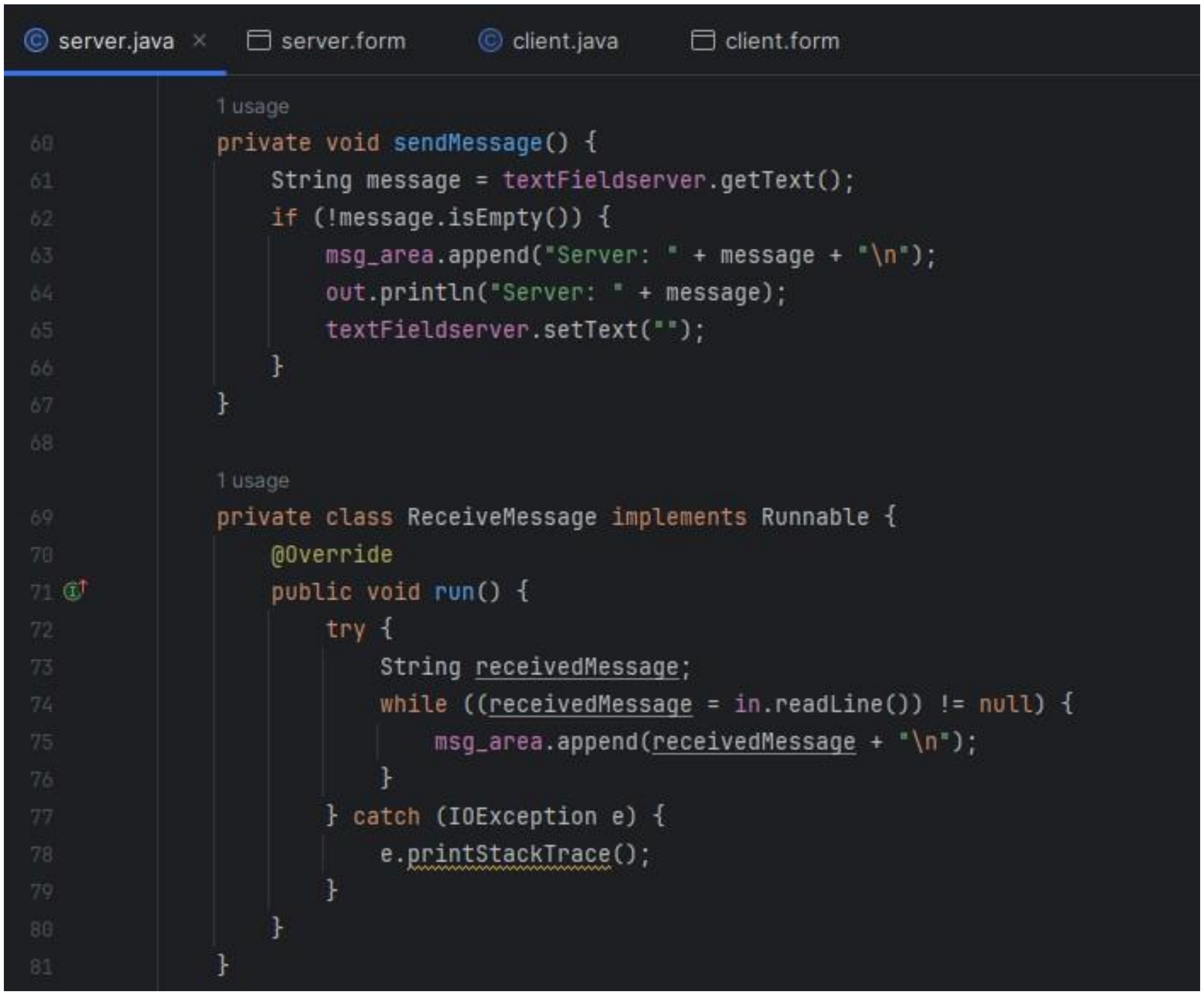
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;

public class server extends JFrame {
    7 usages
    private JTextArea msg_area;
    5 usages
    private JTextField textFieldserver;
    4 usages
    private JButton buttonserver;

    2 usages
    private ServerSocket serverSocket;
    3 usages
    private Socket clientSocket;
    2 usages
    private PrintWriter out;
    2 usages
    private BufferedReader in;
```



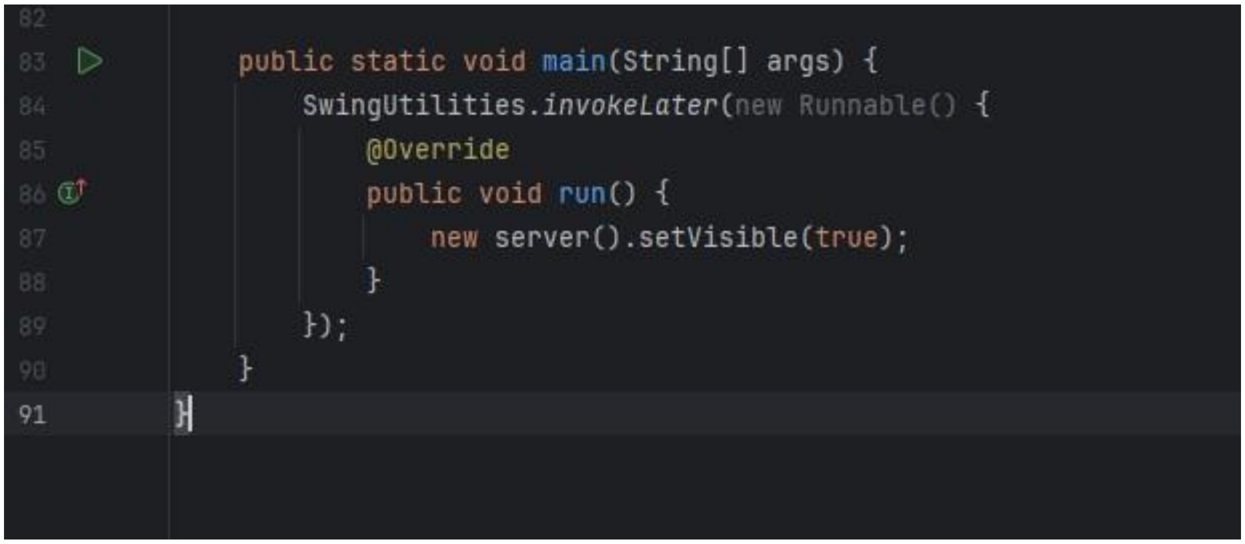
```
    1 usage
    public server() {
        setTitle("Chat Server");
        setSize( width: 400,  height: 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        msg_area = new JTextArea();
        textFieldserver = new JTextField();
        buttonserver = new JButton( text: "Send");

        setLayout(new BorderLayout());
        add(new JScrollPane(msg_area), BorderLayout.CENTER);
        add(textFieldserver, BorderLayout.SOUTH);
        add(buttonserver, BorderLayout.EAST);

        buttonserver.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                sendMessage();
            }
        });

        initializeServer();
    }
```


server.java × server.form client.java client.form

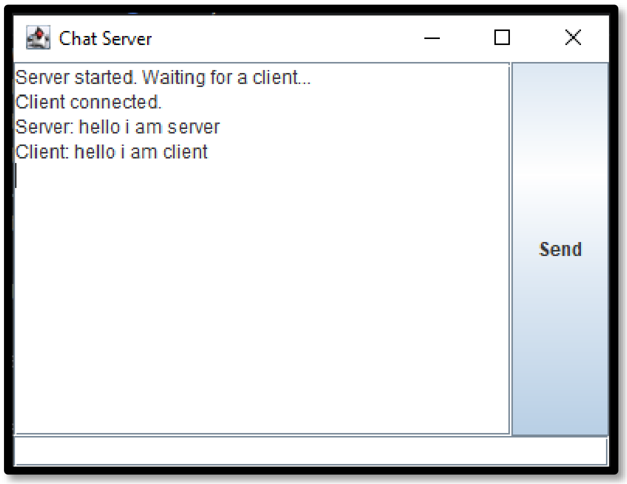
```java
    1 usage
    private void initializeServer() {
        try {
            serverSocket = new ServerSocket( port: 12345);
            msg_area.append("Server started. Waiting for a client...\n");

            clientSocket = serverSocket.accept();
            msg_area.append("Client connected.\n");

            out = new PrintWriter(clientSocket.getOutputStream(),  autoFlush: true);
            in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));

            new Thread(new ReceiveMessage()).start();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```


server.java × server.form client.java client.form

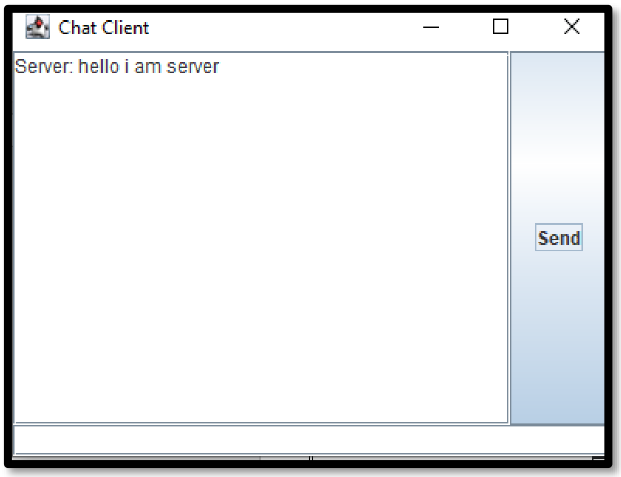
```java
    1 usage
    private void sendMessage() {
        String message = textFieldserver.getText();
        if (!message.isEmpty()) {
            msg_area.append("Server: " + message + "\n");
            out.println("Server: " + message);
            textFieldserver.setText("");
        }
    }

    1 usage
    private class ReceiveMessage implements Runnable {
        @Override
        public void run() {
            try {
                String receivedMessage;
                while ((receivedMessage = in.readLine()) != null) {
                    msg_area.append(receivedMessage + "\n");
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
```


```java
    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                new server().setVisible(true);
            }
        });
    }
}
```

OUTPUT:



```
Server started. Waiting for a client...
Client connected.
Server: hello i am server
Client: hello i am client
```



```
Server: hello i am server
```